1 Energy Function Motivation

Lots of generative models can be represented by probability distributions $p(x \mid w)$ where x is some data/input vector and w is a series of learnable parameters. In order to be a valid probability distribution, we need

$$\int p(x \mid w)\, dx = 1.$$

Consider an arbitrary function $E(x, w)$. In practice, $E(x, w)$ is modeled either through a neural network or another architecture (with parameters w and input x). The exponential $\exp(-E(x, w))$ is a non-negative quantity that can be viewed as an *un-normalized* probability distribution of x; higher energy values correspond to lower probabilities.

We then define

$$p(x \mid w) = \frac{1}{Z(w)} \exp(-E(x, w))$$

where

$$Z(w) = \int \exp(-E(x, w))\, dx.$$

(a) Given a dataset of i.i.d. samples $\mathcal{D} = \{x_1, x_2, \ldots, x_N\}$, we would like to compute the log-likelihood $\log p(\mathcal{D} \mid w)$, so that we can calculate gradients later. In terms of $E(x_i, w)$ and $Z(w)$, what is the log-likelihood $\log p(\mathcal{D} \mid w)$?

(b) From the above we can calculate

$$\mathbb{E}_{x \sim p_{\mathcal{D}}}[\nabla_w \log p(x \mid w)] = -\mathbb{E}_{x \sim p_{\mathcal{D}}}[\nabla_w E(x, w)] - \nabla_w \log Z(w),$$

since the samples x_i are i.i.d. from some distribution $p_{\mathcal{D}}$. Note that the second term doesn't depend on $\mathcal{D}$ and only depends on the normalizing function $Z(w)$.

Show that $-\nabla_w \log Z(w) = \int \nabla_w E(x, w) p(x \mid w)\, dx$.

(c) The above shows that $-\nabla_w \log Z(w) = \mathbb{E}_{x \sim p(x|w)}[\nabla_w E(x, w)]$. Use this result and the result from part (a) to derive a simplified expression for $\mathbb{E}_{x \sim p_{\mathcal{D}}}[\nabla_w \log p(x \mid w)]$. The result you get will form a basis for why we use Langevin sampling to help approximate this gradient.

(d) How can we use Langevin sampling to help approximate the gradient?

2 (un-adjusted) Langevin Sampling

We'll now go through a concrete example of applying Langevin sampling when the target distribution is Gaussian. We'll analyze the convergence rate of this algorithm as well as the actual distribution it converges to.

Recall that the Langevin update equation is given by

$$x_{t+1} = x_t + \eta \nabla_x \log p(x_t) + \sqrt{2\eta} \cdot \epsilon_t$$

where $\epsilon_t \sim N(0, I)$, $\nabla_x \log p(x_t)$ is a *score function*, and $\eta > 0$ is the step size.

This looks like a "noisy" version of gradient ascent where we add noise in every step. While gradient ascent will always find a local maxima (given appropriate step sizes), this algorithm often converges upon somewhere near a local maxima but due to the stochastic nature sometimes ends up in smaller probability regions.

For the rest of this question, assume

$$f(x) = \frac{1}{2}(x - \mu)^\top H(x - \mu)$$

We're going to apply Langevin sampling to this function to show that we eventually converge to a sample from $p(x) = \frac{\exp(-f(x))}{Z}$ where $Z = \int \exp(-f(x))\, dx$. Note that $p(x)$ is a Gaussian distribution centered at μ with covariance matrix H^{-1}.

(a) Derive $\nabla_x \log(p(x))$.

(b) Rewrite the score function using the gradient above. What is $x_t - \mu$ in terms of $x_0 - \mu$?

(c) What is the expectation of x_t? What does this approach as $t \to \infty$? Assume that the eigenvalues of $I - \eta H$ are upperbounded in absolute value by 1 (i.e. η is set appropriately small).

(d) To simplify calculations, assume $H = I$ (i.e. the covariance of $p(x)$ is just the identity matrix). What is the covariance of x_t, i.e., $\mathbb{E}[(x_t - \mu)(x_t - \mu)^\top]$ as $t \to \infty$? What do you notice about this? As $\eta \to 0$ what does this approach?